This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Object Recognition as Next Token Prediction

Kaiyu Yue[12*] Bor-Chun Chen[1] Jonas Geiping[3] Hengduo Li[1] Tom Goldstein[2] Ser-Nam Lim[4]
[1]Meta [2]University of Maryland [3]ELLIS Institute & MPI-IS Tübingen [4]University of Central Florida

Abstract

We present an approach to pose object recognition as next token prediction. The idea is to apply a language decoder that auto-regressively predicts the text tokens from image embeddings to form labels. To ground this prediction process in auto-regression, we customize a non-causal attention mask for the decoder, incorporating two key features: modeling tokens from different labels to be independent, and treating image tokens as a prefix. This masking mechanism inspires an efficient method – one-shot sampling – to simultaneously sample tokens of multiple labels in parallel and rank generated labels by their probabilities during inference. To further enhance the efficiency, we propose a simple strategy to construct a compact decoder by simply discarding the intermediate blocks of a pretrained language model. This approach yields a decoder that matches the full model's performance while being notably more efficient. The code is available at `github.com/kaiyuyue/nxtp`.

1. Introduction

This paper delves into a fundamental problem in computer vision – object recognition – translating an image into object labels. Generally speaking, the recognition framework comprises an image encoder and a decoder. The image encoder, either in the form of a convolutional neural network (CNN) [43, 60, 72, 106, 110] or a vision transformer (ViT) [28, 93, 120], produces image embeddings, while the decoder propagates them to predict object labels.

If the decoder is a linear classifier [28, 43, 60, 72, 106, 110], it needs to be initialized with fixed object concepts. ResNet [43], for instance, initializes its final linear layer with 1K embeddings, a.k.a. weights, to represent 1K objects in ImageNet [25]. Such static weights, however, limit the model's ability to recognize any object. This limitation can be mitigated using a language model [26, 114] as the decoder to generate a flexible set of object embeddings from input descriptions. For example, CLIP [93] encodes the object descriptions into dynamic weights by prompting with "a photo of a $\{\mathcal{L}\}$", where $\mathcal{L}$ could be any object name, and matches these weights with image embeddings to recognize objects.

*Work done during an internship at Meta AI. `kaiyuyue@cs.umd.edu`.

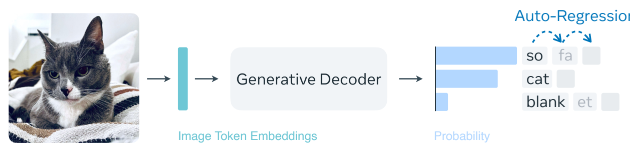


Figure 1. **Object recognition as next token prediction** using a generative decoder such as a transformer-based language model to auto-regressively predict object labels. *Photo authorized with CC BY 4.0.*

Note that CLIP predefines the gallery with a fixed number of object descriptions prior to inference. This requirement reveals that CLIP's object embeddings cover only a portion of the textual space in practical scenarios, rather than its entirety. Additionally, enlarging the gallery has been shown to diminish its performance [19]. Given these observations, a question arises: Can we eliminate the predefined object labels or descriptions?

A direct strategy could use a generative model, particularly a large language model (LLM) [11, 87, 91, 92, 112–114], to decode labels from image embeddings. For instance, Flamingo [1, 3] employs a LLM to transform image embeddings into textual outputs for various vision tasks such as object recognition, image captioning, and visual question answering (VQA). But producing the desired results for a specific task needs several reference samples as few-shot prompts for the model. In other words, it requires predefined reference pivots to refine and align its predictions more precisely with the target task.

The most straightforward alternative is to skip any predefining procedure and align the LLM with the recognition task directly. This approach hinges on the fact that a LLM's token embeddings represent the entire textual space, including all object labels. This is as opposed to predefining subsets, i.e., query galleries or reference pivots, of this space that potentially constrains the model's capability.

Building on this concept, we propose a simple method that employs a language decoder to auto-regressively decode object labels token-by-token from image embeddings, as depicted in Figure 1. We operate a pretrained CLIP image encoder [93] to produce image embeddings, already aligned with text, and linearly transform them to match the language decoder's embedding dimension.

This auto-regressive framework, unlike the contrastive framework exemplified by CLIP [93], is trained to predict text embeddings from image embeddings, rather than aligning both. While related in spirit to recent vision-language models such as LiMBeR [81], LLaVA [68, 69], and BLIP-2 [64, 65], our method introduces differences and innovations:

First, our approach targets object recognition, as opposed to the chat-oriented VQA methods. We train on image-caption pairs, easier to collect and annotate than image-question-answer triplets, and extract nouns from raw captions as reference labels to weakly supervise training. For inference, we generate text fragments as labels rather than sentences. In scenarios like recommendation systems [97] that require labels or tags, a simple label-only output is more concise than verbose sentences requiring further post-processing.

Second, our decoder has a different token modeling mechanism. Instead of decoding all input and output tokens in a conditional sequence as in LLMs, we ensure tokens from different labels to be independent, while tokens from the same label remain conditional. Naturally, all label tokens are conditional on image embeddings. This decoupling is based on the understanding that different labels in the same image are independent but their coexistence is determined by the underlying visual context. To this end, we customize a non-causal attention mask for our language decoder.

Further, the non-causal masking mechanism inspires a new sampling method, called *one-shot sampling*, to generate text tokens for labels. Instead of sampling tokens in sequence as in greedy search, beam search, and nucleus sampling [50], one-shot sampling simultaneously samples tokens of multiple labels in parallel and ranks them by their probabilities. This makes use of the strong parallelization capabilities of a transformer, leading to object recognition that is much more efficient than the aforementioned methods and does not suffer from repetition issues [35, 121].

Lastly, we put forth a straightforward strategy to enhance model efficiency of our recognition model. We hypothesize that only partial knowledge in LLMs is vital for recognition and focus on maximizing efficiency by not engaging the entire language model. To construct the decoder, we start with a pretrained LLM, e.g., LLaMA [112, 113], retain first six transformer blocks along with the final output layer, and drop the intervening blocks. This compact decoder matches the full model's performance but is substantially more efficient, i.e., 4.5× faster in inference.

2. Related Work

Aligning Images and Text, including sentences, phrases, or words, in a shared space has been prevalent for image-text matching [9, 23, 34, 49, 57, 59, 78, 108, 119], and foundational in contrastive frameworks [40, 75, 93], while others are geared towards generating text descriptions from images [55, 56, 59, 78, 108, 115]. Then, integrating visual perception with LLMs [114] like GPT [11, 87, 91, 92] and LLaMA [112, 113] is gaining traction by treating image embeddings as language token embeddings, seamlessly fusing visual and textual information within the model [48, 105]. Such methods are being applied to tasks such as detection [14], few-shot recognition [1, 93], textual explainations [10], classification justification [45], bottleneck models [100, 122], reasoning [2, 42, 46, 77, 80, 103], and chat-based models [22, 64, 65, 68, 69, 81] for captioning and VQA.

Tackling Open-Vocabulary Tasks for recognition [93], detection [29, 38, 61, 82, 83, 123] and segmentation [29, 36] typically involves training on a set of base labels and then recognizing rare unseen labels. The cornerstone of open-vocab approaches is the contrastive learning [41, 109] like CLIP [93], which employs a language model to encode labels to contrast with images. Therefore, open-vocab methods potentially inherit CLIP's limitations discussed in Section 1 due to the predefined base and rare labels. CaSED [19] utilizes raw captions to form a vocabulary-free gallery, diverging from the gallery of predefined label vocabularies. However, its performance is heavily dependent on gallery selection, as demonstrated in Table 10 of [19], highlighting its limitations as a retrieval-based method.

We argue that by dramatically increasing the training data to cover a wide array of objects, the reliance on recognizing rare data and concepts can be heavily reduced. Our method aligns more with the open-world paradigm [6] that incrementally learns new labels over time, mirroring the way of data collection in the real world. In the application, given just an image, our model predicts labels with ranking probabilities, without relying on any predefined set of concepts.

3. Method

3.1. Revisiting Object Recognition

We begin by briefly reviewing object recognition in its general formulation. Suppose that 2D images are fed into a backbone, e.g. ViT [28] in CLIP [93], which produces image embeddings[1] $\mathbf{X}_\text{v} \in \mathbb{R}^{M\times D}$, where M is the spatial size and D is the embedding dimension. In a nutshell, the problem of recognition aims to decode object labels solely from $\mathbf{X}_\text{v}$, translating image embeddings into the textual space.

In the past years, the core design of this translation employs a set of textual embeddings $\mathbf{W} \in \mathbb{R}^{N\times D}$ to seek the optimal alignment with $\mathbf{X}_\text{v}$:

$$\arg\max\ \sigma(\mathbf{W} f(\mathbf{X}_\text{v})^\top), \tag{1}$$

[1]Bold capital letters denote a matrix $\mathbf{X}$, and bold lower-case letters a column vector $\mathbf{x}$. $\mathbf{x}_i$ and $\mathbf{x}^j$ represents the i^th row and j^th column of the matrix $\mathbf{X}$ respectively. $\mathbf{X}_{ij}$ denotes the scalar in the i^th row and j^th column of the matrix $\mathbf{X}$. All non-bold letters represent scalars.

where σ is the softmax function and f is to transform $\mathbf{X}_\text{v}$ for aligning with $\mathbf{W}$. For instance, linear classifiers such as ResNet [43] employ the average pooling as f to transform $\mathbf{X}_\text{v}$ to a single vector representation, and initiate $\mathbf{W}$ using a set of predefined concepts corresponding to object labels, e.g., $N = 1000$ for ImageNet [25]. The contrastive frameworks such as CLIP [93] embed a collection of predefined object descriptions into $\mathbf{W}$, and apply an aggregation (like [CLS] embedding [28]) and linear projection as f on $\mathbf{X}_\text{v}$.

Eq. 1 aims to maximize the alignment between $f(\mathbf{X}_\text{v})$ and $\mathbf{W}$. The space of $\mathbf{W}$ plays a critical role in this alignment as the diversity and richness of the embeddings in $\mathbf{W}$ directly affect the model's ability to differentiate objects. The linear classifiers and contrastive frameworks, however, limit $\mathbf{W}$ to a predefined subset that potentially constrains the model's capability to recognize any object. Our goal is to eliminate this limitation and extend $\mathbf{W}$ to the entire textual space.

3.2. Auto-Regression for Recognition

Recently, LLMs have significantly advanced in understanding and generating text [11, 87, 91, 92, 112–114]. Considering that their token embeddings are trained to represent the entire textual space, we define $\mathbf{W}$ with the token embeddings[2] from a pretrained LLM, e.g., LLaMA [112, 113], featuring $N = 32$K textual tokens. Then Eq. 1 changes to predicting the token:

$$P(\mathbf{w}|\mathbf{X}_\text{v}) = \arg\max \, \sigma(\mathbf{W} f(\mathbf{X}_\text{v})^\top), \tag{2}$$

where $\mathbf{w}$ represents the most probable single token for $\mathbf{X}_\text{v}$. In our method, f is a combination of linear projection and the pretrained LLM to project $\mathbf{X}_\text{v}$ in the textual space of $\mathbf{W}$. That is, f is our language decoder.

To guide the language decoder in the recognition task, we prompt it with a short instruction – "the objects in the image are" – tokenized as $\mathbf{X}_\text{p} \in \mathbb{R}^{P \times D}$. Then we concatenate $\mathbf{X}_\text{v}$ and $\mathbf{X}_\text{p}$ to form our input token embeddings:

$$\mathbf{X} = \mathbf{X}_\text{v} \oplus \text{[IMG]} \oplus \mathbf{X}_\text{p}, \tag{3}$$

where $\oplus$ is the concatenation operation and [IMG] is a special token to indicate the boundary.

Typically, a label consists of multiple tokens, e.g., "sofa" has two tokens [so] and [fa]. Without loss of generality, we assume a label L has T tokens. Now predicting L is equivalent to auto-regressively predicting its tokens:

$$P(L) = P(\mathbf{w}_1, \ldots, \mathbf{w}_T|\mathbf{X}_\text{v}, \mathbf{X}_\text{p}) = \prod_{t=1}^{T} P(\mathbf{w}_t|\mathbf{w}_{<t}, \mathbf{X}), \tag{4}$$

[2] In general, LLMs have two sets of token embeddings, one for encoding input tokens and the other for predicting output tokens. Some LLMs like GPT-2 [92] share the same embeddings for both input and output tokens [90], while others like LLaMA [113] employ different embeddings. Our method defines $\mathbf{W}$ with the embeddings designated for output tokens.

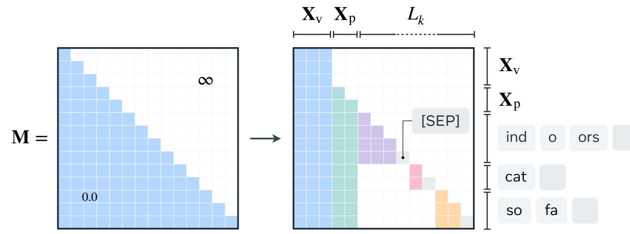


Figure 2. **Non-causal attention mask** for prefixing image tokens $\mathbf{X}_\text{v}$ and decoupling tokens from different labels L_k to be independent at the [SEP] token.

where $\mathbf{w}_t$ is the t-th token of L, and $\mathbf{w}_{<t}$ is the sequence of tokens before the t-th token. To compute the conditional probability in Eq. 4, the transformer-based LLM in f employs a causal mask $\mathbf{M}$ [114] on the pairwise attention $\mathbf{A}$ to model the interdependence between tokens:

$$\mathbf{A} \leftarrow \mathbf{A} + \mathbf{M}, \quad \mathbf{M} = \text{tril}(\infty), \tag{5}$$

where tril(∞) is with zeros in the lower triangle and infinity values in the upper triangle. This enforces the token $\mathbf{w}_t$ to attend only to the preceding tokens $\mathbf{w}_{<t}$, i.e., making $\mathbf{w}_t$ conditional on $\mathbf{w}_{<t}$, as shown in the left of Figure 2.

3.3. Non-causal Masking

In general, an image contains multiple objects, and our goal is to predict them all. Suppose there are K objects, and we denote the output set of labels for the image as $\mathcal{L} = \{L_1, ..., L_K\}$, where k-th label has $T_k + 1$ tokens, including the special token [SEP] for the delimiter. Then the likelihood of this set of labels appearing in the image is the product of their probabilities:

$$P(\mathcal{L}) = \prod_{k=1}^{K} P(L_k) = \prod_{k=1}^{K} \prod_{t=1}^{T_k+1} P(\mathbf{w}_t^k|\mathbf{w}_{<t}^k, \mathbf{X}). \tag{6}$$

Now Eq. 6 is not a standard auto-regression practiced in LLMs because $\mathbf{w}_t^k$ only needs to attend to the input tokens $\mathbf{X}$ and the preceding tokens $\mathbf{w}_{<t}^k$ from the same label L_k. This is supported by the understanding that the labels coexist in the same image due to the underlying visual context, but are independent of each other. Additionally, the image tokens $\mathbf{X}_\text{v}$ exhibit inherently spatial correlation, in contrast to the temporal correlation of natural language tokens. Therefore, we customize a non-causal attention mask $\mathbf{M}$ with two designs, illustrated in the right of Figure 2: a) We decouple the correlation between tokens from different labels at the [SEP] token to prevent these tokens from being attended to each other; b) We treat image tokens $\mathbf{X}_\text{v}$ as a prefix [27, 70, 94, 116–118], enabling the image tokens to *see* each other. Interestingly, our non-causal attention mask shares a similar design as the column mask in [95] but is developed from a different perspective, where the column mask is specifically for image-to-image attention.

In the end, Eq. 6 is our final training objective. We use the cross-entropy loss for optimization, with weakly-supervised labels[3] $\mathcal{L}$ extracted from the corresponding image captions.

3.4. One-Shot Sampling

The non-causal masking decouples the tokens from distinct labels, indicating that the first token of any label could be the next after $\mathbf{X}$ in the first sampling round. In other words, a higher probability for the first token, being sampled after input $\mathbf{X}$, would result in a higher relevance of the label to the image. This inspires us to sample tokens of multiple labels in parallel, as shown in Figure 3.

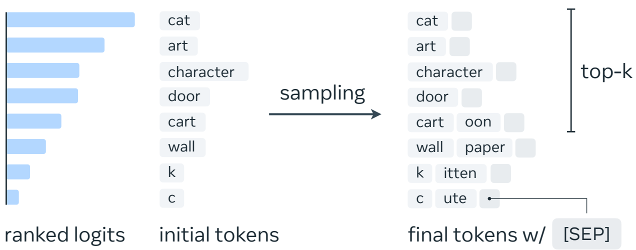


Figure 3. **One-shot sampling** for generating tokens of top-k labels in parallel. Once the model samples the [SEP] token, the label is completed. Otherwise, the model continues for unfinished labels.

Given input tokens $\mathbf{X}$, we propagate them into the decoder and rank the output logits by their softmax probabilities. The top-k tokens, called initial tokens, decide the top-k labels to be generated. The efficacy of linking initial tokens to final labels is explored in Table 8, highlighting the promise of this straightforward approach. Then we sample the next token for the top-k initial tokens in parallel, using top-1 sampling, to generate k labels. If the sampled token is [SEP], the label is completed. Otherwise, the model continues to sample the next token for the unfinished labels. Finally, we report the probability of each label as the product of its token probabilities. We refer to this approach as *one-shot sampling,* which enables parallel sampling of multiple labels in one shot. The key to its parallelism lies in the non-causal masking mechanism, which also avoids the repetition issue [35, 121] typically faced in greedy and beam search, as it causes the model to focus uniformly on the same input tokens $\mathbf{X}$ across various labels.

To sum up, the one-shot sampling differs from other sampling methods in two essential aspects: a) It operates in parallel across multiple object labels, with each parallel branch processing a small number of tokens (roughly less than ten tokens), in contrast to the sequential sampling of other methods; b) It naturally aligns with the vision recognition task by representing the image as a spatially correlated entity, while other sampling methods depict the image as a sequence of tokens.

[3]Our learning approach is considered weakly-supervised as the labels are incomplete and imperfect derived from raw captions.

3.5. Truncating the Decoder

Now, considering the language model LLaMA in our decoder f, we posit that a specific subset of language understanding in its numerous parameters is vital for recognition. This realization prompts us to focus on maximizing efficiency by not engaging the entire model. We construct our language decoder, initially based on the LLaMA 7B (version 1 or 2), by truncating it to the first 6 transformer blocks along with the final output layer, as depicted in Figure 4, while preserving its tokenizer and the pretrained 32K token embeddings for encoding the input. We designate this modified version as the truncated language decoder, denoted as $\text{Lang}_{\text{truncated}}$ in our experiments.

Figure 4. **Encoder and truncated decoder**. We retain the first 6 transformer blocks along with the final output layer of the LLaMA 7B as our truncated decoder, and train with partial encoder blocks.

4. Experiments

Data. We construct training datasets at two different scales for experiments. **G3M**: a training group of 3M(illion) pairs combines CC3M [104], COCO Captions [15, 67], SBU [88], which is mainly used for ablation studies. **G70M**: We gather 67M pairs from LAION-Synthetic-115M (slightly fewer than previous work due to missing URLs) [64, 102]. Combining it with G3M, we form a 70M-pair training group for scaling-up training. For evaluation, we use the validation split of CC3M, COCO Captions, and OpenImages V7 [7]. We parse the *raw captions* to obtain meaningful nouns as reference labels in both training and evaluation. The processing details are described in Section A.5.

Implementation. The inference augmentation for input images in CLIP [93] is applied in both training and evaluation. The input size is 224^2. The image encoder is ViT-L/14 [28] pretrained from CLIP [93], producing 256 token embeddings with the dimension of 1024, as $\mathbf{X}_\text{v}$. Note that we drop its [CLS] token. The special token embedding of [IMG] is learned during training. The special token [SEP] is the comma (,), and 32K token embeddings for the input are fixed. The max number of input tokens is 512. No [EOS] token, i.e., the end of the sentence, is used in the input. We shuffle labels for each image in training.

Training. AdamW [74] with the cosine annealing learning rate (LR) schedule [73] is applied in single-stage training. The multi-dimensional parameters apply a weight decay of 10^{-1}. The global batch size is 512 with 32 NVIDIA A100-SXM4-80GB GPUs. The warm-up has 2K iterations. We jointly train four parts: the last 6 blocks of the image encoder ViT-L/14, the projection layer for $\mathbf{X}_\text{v}$, the special

[IMG] token embedding, and the whole truncated language decoder, using a LR of 10^{-5} for 3 epochs, as shown in Figure 4, taking ∼5 hours on G3M and ∼5 days on G70M.

Evaluation. The n-gram overlap metrics, including BLEU [89] and ROUGE [66], are widely used to evaluate the quality of sentences generated by language models. However, these metrics are not suitable for evaluating the quality of results in recognition tasks. For example, "car" and "automobile" have the low n-gram similarity but are semantically alike. To quantify the semantic similarity between the generated labels and the reference labels, we adopt the concept from BERTScore [124] to formulate our evaluation metric[4].

Formally, given a set of reference labels $\mathcal{R}$ with size M and a set of generated labels $\mathcal{G}$ with size N, we use the sentence-BERT [96] to encode $\mathcal{R}$ to a set of semantic embeddings $\mathbf{R} \in \mathbb{R}^{M \times D}$ and $\mathcal{G}$ to $\mathbf{G} \in \mathbb{R}^{N \times D}$, where D is the embedding dimension. Then we compute the cosine similarity matrix $\mathbf{S} \in \mathbb{R}^{M \times N}$ between $\mathbf{R}$ and $\mathbf{G}$:

$$\mathbf{S}_{ij} = \frac{\mathbf{r}_i\, \mathbf{g}_j^\top}{\|\mathbf{r}_i\| \|\mathbf{g}_j\|} \in \mathbb{R}^{[-1,1]}. \tag{7}$$

We compute the recall for the reference set $\mathbf{R}$ and the precision for the generated set $\mathbf{G}$:

$$R = \frac{1}{M} \sum_{i=1}^{M} \max_j \mathbf{S}_{ij}, \quad P = \frac{1}{N} \sum_{j=1}^{N} \max_i \mathbf{S}_{ij}, \tag{8}$$

where $\max$ indicates the greedy matching strategy following [124]. Finally, we compute the F_1 score as the harmonic mean of R and P:

$$F_1 = \frac{2RP}{R + P}. \tag{9}$$

For each sample, we evaluate the top-k generated labels out of N and report the average R, P, and F_1 over all samples.

Note that, different models may have different numbers of generated labels N for each image. Especially, when $N < k$, we do not pad the matrix $\mathbf{S}$ with zeros to make $N = k$ and penalize the model. Thus, the model with $N < k$ will have a higher P compared to the model with $N = k$.

4.1. Main Results

The comprehensive comparisons with other related methods, including CLIP [93], Open Flamingo [3], LLaVA [68, 69], BLIP-2 [65], InstructBLIP [22], and CaSED [19], are detailed in Table 1 with top-10 predictions, and Table A.10 with top-5 predictions.

Preliminary. We construct two galleries for CLIP: a) the base gallery, highlighted in gray, contains reference labels only from the corresponding test dataset, e.g., CC3M validation labels for CC3M evaluation. b) the extended gallery,

[4]The metric essentially measures the model's accuracy, as explained in Section A.4.

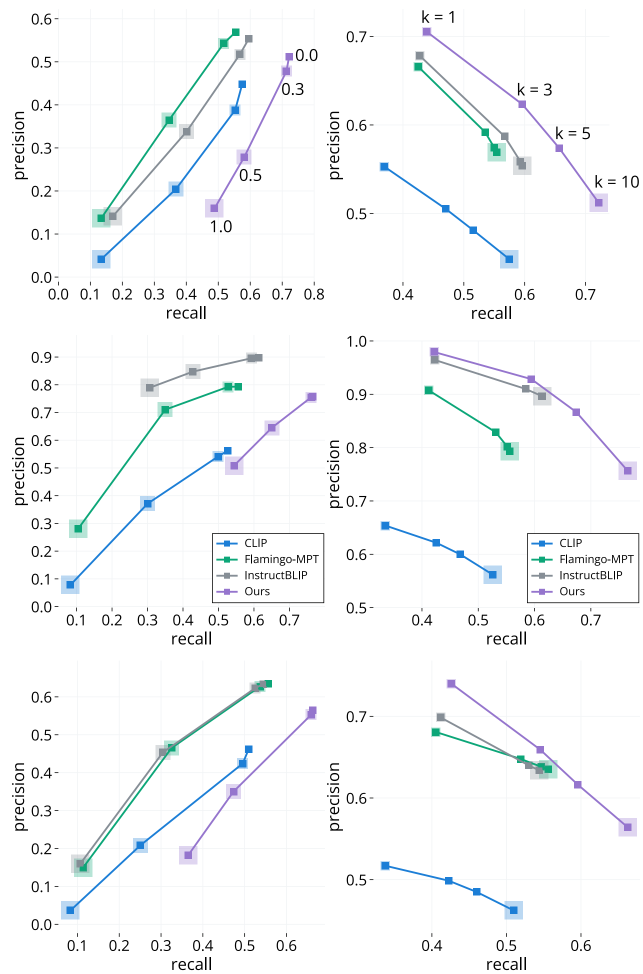


Figure 5. **Precision-recall (PR) curves** on CC3M, COCO, and OpenImages validation splits within 3 rows from top to bottom. The left column is the PR curves with different thresholds, i.e., $[0.0, 0.3, 0.5, 1.0]$, applying on the similarity matrix $\mathbf{S}$ in Eq. 7. The right column is the PR curves with different top-k predictions, where k is $[1, 3, 5, 10]$. All figures share the same legend.

includes all reference labels from the G3M training group.

Regarding CaSED [19], its performance is significantly impacted by the search gallery composition. For a fair comparison, we evaluate CaSED using: a) the released gallery provided with the paper, in gray, featuring CLIP ViT-L/14 text embeddings from CC12M [104]; b) the extended gallery, comprising CLIP ViT-L/14 text embeddings from COCO, SBU, CC3M, and LAION-400M, which covers our G70M training group. CaSED can be considered a CLIP variant, with its defining aspect being the enhanced query gallery.

We evaluate other methods using their largest publicly available models. We employ two prompt types, *list* and *caption*, to generate object labels from them, detailed in Section A.6. Also, we use the *instruct* prompt for instruction-based methods, similar to its use for GPT-4V Preview [86] in A.1.

method	models (vision + lang)	prompt	data scale	# params (B)	CC3M R	CC3M P	CC3M F_1	COCO R	COCO P	COCO F_1	OpenImages R	OpenImages P	OpenImages F_1
CLIP [93]	ViT L-14 + $CLIP_{lang}$	-	400M	0.43	0.575	0.448	0.499	0.525	0.562	0.540	0.510	0.462	0.480
CaSED [19]	ViT L-14 + Retrieval	-	12M	0.43	0.648	0.471	0.540	0.582	0.592	0.584	0.534	0.470	0.494
CLIP [93]	ViT L-14 + $CLIP_{lang}$	-	400M	0.43	0.451	0.383	0.409	0.429	0.483	0.450	0.386	0.363	0.371
CaSED [19]	ViT L-14 + Retrieval	-	403M	0.43	0.653	0.481	0.548	0.616	0.629	0.620	0.560	0.494	0.519
$Flamingo_{open}$ [3]	ViT L-14 + LLaMA 1 [112]	list	2.1B	8.34	0.547	0.540	0.536	0.549	0.721	0.618	0.526	0.621	0.562
$Flamingo_{open}$	ViT L-14 + LLaMA 1	caption	2.1B	8.34	0.548	0.521	0.527	0.553	0.697	0.611	0.538	0.607	0.563
$Flamingo_{open}$	ViT L-14 + MPT [111]	list	2.1B	8.13	0.554	<u>0.569</u>	0.553	0.556	0.793	0.646	0.555	0.635	0.584
$Flamingo_{open}$	ViT L-14 + MPT	caption	2.1B	8.13	0.534	0.533	0.527	0.554	0.754	0.633	0.551	0.613	0.574
$LLaVA_{1.0}$ [69]	ViT L-14 + LLaMA 2 [113]	list	753K	13.3	0.540	0.528	0.526	0.580	0.803	0.666	0.543	0.641	0.580
$LLaVA_{1.0}$	ViT L-14 + LLaMA 2	caption	753K	13.3	0.634	0.460	0.528	0.688	0.668	0.675	0.610	0.511	0.550
$LLaVA_{1.0}$	ViT L-14 + LLaMA 2	instruct	753K	13.3	0.588	0.450	0.505	0.638	0.631	0.632	0.615	0.541	0.570
$LLaVA_{1.5}$ [68]	ViT L-14 + Vicuna [16]	list	1.2M	13.4	0.538	0.515	0.518	0.591	0.783	0.665	0.552	0.614	0.574
$LLaVA_{1.5}$	ViT L-14 + Vicuna	caption	1.2M	13.4	0.632	0.453	0.522	0.679	0.649	0.661	0.611	0.508	0.549
$LLaVA_{1.5}$	ViT L-14 + Vicuna	instruct	1.2M	13.4	0.572	0.498	0.522	0.630	0.716	0.659	0.615	0.577	0.582
BLIP-2 [65]	ViT g-14 + Flant5xxl [17]	list	129M	12.2	0.544	0.557	0.542	0.494	0.871	0.623	0.476	<u>0.641</u>	0.538
BLIP-2	ViT g-14 + Flant5xxl	caption	129M	12.2	0.600	0.539	0.561	0.600	<u>0.893</u>	0.714	0.523	0.626	0.561
InstructBLIP [22]	ViT g-14 + Flant5xxl	list	129M	12.3	0.596	0.554	0.567	0.613	**0.897**	<u>0.725</u>	0.544	0.634	0.578
InstructBLIP	ViT g-14 + Flant5xxl	caption	129M	12.3	0.639	0.487	0.546	0.690	0.662	0.673	<u>0.647</u>	0.539	0.581
InstructBLIP	ViT g-14 + Flant5xxl	instruct	129M	12.3	0.529	**0.604**	0.555	0.569	0.879	0.686	0.561	**0.698**	**0.615**
Ours	ViT L-14 + $Lang_{truncated}$	-	3M	1.78	**0.738**	0.530	**0.611**	<u>0.700</u>	0.712	0.702	0.613	0.544	0.570
Ours	ViT L-14 + $Lang_{truncated}$	-	70M	1.78	<u>0.722</u>	0.512	<u>0.593</u>	**0.765**	0.757	**0.758**	**0.663**	0.564	<u>0.603</u>

Table 1. **Comparison of different methods with top-10 predictions**. Bold numbers are the best results and underlined numbers are the second best results, same for the following tables.

Analytic Comparisons. In the R column of Table 1, R remains consistent as the number of reference labels per sample is fixed, so unaffected by prediction count. Higher R suggests top-k predictions have higher semantic relevance to the reference labels. Our method outperforms others for top-10 predictions across all datasets, showing our approach's ability to yield more relevant labels.

The P column is sensitive to the quantity of predictions; for instance, if we assess top-10 predictions but the model produces only five labels, the precision will be higher than that of the model yielding 10 predictions, according to Eq. 8. To better understand the P/R relationship, we plot two different precision-recall (PR) curves in Figure 5, calculated by adjusting the match threshold between references and predictions, and altering k for predictions.

The left column of Figure 5 derives from various thresholds on the similarity matrix $\mathbf{S}$ in Eq. 7 with top-10 predictions. The curves demonstrate a strong linear correlation due to the calculation of P and R from the best matches in $\mathbf{S}$. A threshold of 0.7, for example, excludes pairs with lower similarity, reducing both P and R simultaneously. The rate at which P and R decline with increasing thresholds reflects the overall similarity of predictions to reference labels – a faster drop means the lower overall similarity. Our method, with the gradual descent of the curves, suggests better prediction quality across all test datasets. At a threshold of 1.0, non-zero values of P and R signify that the model's predictions perfectly match the reference labels.

The right column of Figure 5 shows the PR curves for varying top-k predictions, with the inverse correlation between P and R, indicating their trade-off. Our method outperforms others in both P and R at top-1 and -3, while at top-5, $Flamingo_{open}$ and InstructBLIP saturate at the same level as top-10, even we double their sampling tokens for trying to generate more. This observation demonstrates that VQA-based models are suboptimal for the task due to the lack of the ability to generate diverse labels consistently. The plateau explain their highest P, but lower R and F_1 in Table 1. Our method can achieve higher recall with increasing k, showing that it can consistently hold a P/R balance.

5. Ablation Studies

Truncating the Language Decoder. To test our conjecture that only a subset of knowledge in LLMs is vital for the task, we reduce the decoder's size starting from LLaMA 7B. We have found that removing intermediate transformer blocks results in a compact decoder with comparable performance.

To begin, we need to determine which transformer blocks to remove out of the 32 blocks in LLaMA 7B. Drawing inspiration from [44], we initially fine-tuned the last third, i.e., 11 blocks, along with the final output layer. On the other hand, motivated by the observation that the language decoder takes image embeddings as the input with a novel domain, we fine-tune the first third of the blocks, i.e., 11 blocks, and the final output layer. This approach is premised on the hypothesis that the initial blocks might be better suited to learn the image embeddings. As evidenced by Table 2, indeed the first third of the LLaMA 7B emerges as the most significant segment. Therefore, we decided to remove blocks after the 11^{th} block.

	CC3M			COCO			OpenImages		
f.t. part	R	P	F_1	R	P	F_1	R	P	F_1
first third	**0.679**	**0.602**	**0.632**	**0.621**	**0.802**	**0.698**	**0.559**	**0.593**	**0.569**
last third	0.651	0.586	0.611	0.585	0.748	0.654	0.550	0.587	0.562

Table 2. **Partial fine-tuning** (f.t.) results of LLaMA 7B with top-5 predictions, sampled by one-shot method. The first third encompasses the first 11 transformer blocks plus the final output layer, while the last third includes the last 11 blocks with the output layer.

	CC3M			COCO			OpenImages		
# params	R	P	F_1	R	P	F_1	R	P	F_1
7.05B - 32	0.679	0.602	0.632	0.621	0.802	0.698	0.559	0.593	0.569
3.00B - 11	0.676	0.600	0.630	0.622	0.805	0.699	0.561	0.598	0.572
1.78B - 6	0.673	0.598	0.627	0.618	0.799	0.695	0.560	0.595	0.570
1.18B - 3	0.670	0.595	0.624	0.615	0.795	0.692	0.558	0.593	0.568
0.77B - 1	0.665	0.590	0.620	0.610	0.790	0.688	0.555	0.590	0.565

Table 3. **Comparison of different language decoder sizes** with top-5 predictions, sampled by one-shot method. The number of parameters counts both the image encoder (0.43B) and the language decoder. It is paired with the number of transformer blocks in our language decoder, e.g., 1.78B model has 6 blocks in the decoder, denoted as 1.78B - 6.

decoder w/	CC3M			COCO			OpenImages		
LLaMA	R	P	F_1	R	P	F_1	R	P	F_1
3B [113]	0.718	0.522	0.599	0.689	0.702	0.693	0.612	0.546	0.571
7B $\rightarrow$ 2.6B	**0.745**	**0.532**	**0.615**	**0.703**	**0.716**	**0.707**	**0.615**	**0.546**	**0.572**

Table 4. **Comparison between truncated decoder and small language model** at equivalent model size with top-10 predictions.

	CC3M			COCO			OpenImages		
sampling	R	P	F_1	R	P	F_1	R	P	F_1
greedy	0.661	**0.604**	0.624	0.606	**0.802**	0.687	0.549	**0.599**	0.565
beam	0.641	0.590	0.608	0.585	0.772	0.663	0.530	0.577	0.546
one-shot	**0.673**	0.598	**0.627**	**0.618**	0.799	**0.695**	**0.560**	0.595	**0.570**

Table 5. **Comparison of different sampling methods** using top-5 predictions. The greedy and beam search sample up to 64 tokens, and takes first five generated labels as predictions.

Note that, we always retain the final output layer of LLaMA for generating the final logits. Initially, we truncate LLaMA 7B at the 11^{th} block, as illustrated in Figure 4, resulting in a 3B model. Table 3 shows that the 3B model matches the full model in performance. To further explore the impact of the decoder size, we truncate the 3B model's decoder by removing its last 5 transformer blocks to produce a 1.78B model and find it still performs comparably to the full model. Until the 0.77B model, which has only one transformer block, the performance has a noticeable drop but small.

The other way to construct the decoder is directly using relative small LLMs, e.g., LLaMA 3B [113]. Table 4 shows our truncated decoder outperforms LLaMA 3B at the same model scale, indicating that truncated decoders can be benefited from the better token embeddings of the larger LLMs. Plus, truncating enables models to flexibly balance accuracy and efficiency across different model scales as in Table 3.

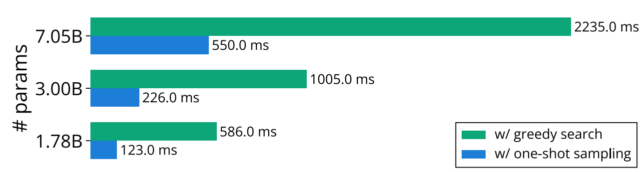
Sampling Strategies. We investigate three deterministic token sampling methods: greedy search, 3-way beam search, and one-shot sampling. Greedy and beam search select the highest probability token, i.e., top-1, at each step. With our model, greedy and beam search suffer from the repetition issue, explained in Section 3.4. To mitigate it for the comparison, we follow [58] to penalize the logits $\mathbf{x}$ of the preceding generated tokens. The sampling distribution for the next token is

$$\mathbf{p} = \frac{\exp(\mathbf{x}_i / (\tau \cdot \mathbb{1}(i \in \mathcal{G})))}{\sum_j \exp(\mathbf{x}_j / (\tau \cdot \mathbb{1}(j \in \mathcal{G})))}, \tag{10}$$

where $\tau = 1.2$ is the penalization factor, $\mathbb{1}(\cdot)$ is the indicator function, and $\mathcal{G}$ is the set of preceding sampled tokens.

The results are shown in Table 5. One-shot sampling considers label count instead of token count in greedy and beam search. It generates more diverse labels without the *repetition issue*, explaining its superior performance in R and F_1 over greedy and beam search, though with marginally reduced P, consistently in top-10 predictions (see Table A.6). Their top-10 comparisons show that, unlike one-shot sampling, increasing the number of tokens in greedy and beam search does not result in more diverse labels.

Note that our one-shot sampling could potentially encounter a *competition issue*, where if multiple plausible labels share the same initial token, it would sample one of them and omit the others. While sampling multiple times for the same token could mitigate this issue, in practice, its impact seems less critical than the repetition issue in sequential sampling. Plus, redundant tokenization can allow multiple labels with the same starting words being returned through different token combinations. This is tentatively indicated by our large-scale predictions in Table 9.



Generation Efficiency. We combine the sampling methods with different decoder sizes to investigate their overall generation efficiency. As illustrated above, the 1.78B model is 4.5× faster than the 7B version in inference. Further, with one-shot sampling and truncated language model, our approach achieves 18.1× speed-up compared to the full model with greedy sampling. The inference time is measured by the average time of generating top-10 labels with one-shot sampling and 64 tokens with greedy search per image. The models run with a batch size of 1 and 16-bit Floating Point, i.e., FP16, on an A100 GPU. Attention is without kv-cache.

Non-causal Masking. In Section 3.3, the non-causal masking considers two aspects: a) prefixing image embeddings

$\mathbf{X}_v$ in the input sequence, and b) decoupling tokens from different labels to be independent. The first ablation is to un-prefix the image embeddings as a sequential input. Table 6 shows that the prefixing is beneficial for the performance, especially with the sequential sampling strategy, i.e., greedy search. For the one-shot sampling, the prefixing helps with a slight improvement on COCO.

The second ablation is to model tokens conditionally from different labels, also shown in Table 6. Independent modeling is able to also provide marginal performance improvement with both greedy search and one-shot sampling, even though it provides significant gains in efficiency due to the parallelized decoding of all object labels.

	CC3M			COCO			OpenImages		
modeling	R	P	F_1	R	P	F_1	R	P	F_1
greedy search									
baseline	0.662	0.577	0.611	0.602	0.754	0.667	0.539	0.559	0.543
+ prefix	0.664	0.580	0.613	0.604	0.759	0.670	0.541	0.563	0.546
+ indep.	0.668	0.600	0.625	0.609	0.797	0.688	0.548	0.588	0.561
one-shot sampling									
baseline	0.677	0.601	0.630	0.611	0.790	0.687	0.556	0.592	0.567
+ prefix	0.678	0.603	0.632	0.613	0.792	0.689	0.557	0.594	0.568
+ indep.	0.679	0.602	0.632	0.621	0.802	0.698	0.559	0.593	0.569

Table 6. **Ablations for prefixing image embeddings and independent modeling of different labels** with top-5 predictions, generated by greedy search and one-shot sampling.

	CC3M			COCO			OpenImages		
version	R	P	F_1	R	P	F_1	R	P	F_1
trained on G3M									
1	0.673	0.598	0.627	0.618	0.799	0.695	0.560	0.595	0.570
2	**0.673**	**0.599**	**0.627**	**0.620**	**0.803**	**0.698**	**0.560**	**0.598**	**0.572**
trained on G70M									
1	**0.659**	**0.576**	**0.609**	**0.674**	**0.866**	**0.755**	**0.594**	**0.615**	**0.597**
2	0.653	0.572	0.604	0.673	0.865	0.754	0.593	0.614	0.596

Table 7. **Comparison of truncating different LLaMA versions** for the language decoder with top-5 predictions.

	CC3M			COCO			OpenImages		
ranking	R	P	F_1	R	P	F_1	R	P	F_1
-	0.673	0.598	0.627	0.618	0.799	0.695	0.560	0.595	0.570
full	0.673	0.598	0.627	0.619	0.800	0.695	0.562	0.597	0.572

Table 8. **Comparison of different strategies for ranking top-5 predictions**. The first row ranks predictions using initial token probabilities, whereas the second row uses full label probabilities, derived by multiplying token probabilities.

	CC3M			COCO			OpenImages		
method	R	P	F_1	R	P	F_1	R	P	F_1
CLIP	0.752	0.360	0.483	0.715	0.430	0.536	0.666	0.387	0.485
CLIP	0.615	0.332	0.427	0.576	0.411	0.478	0.506	0.334	0.399
ours	**0.868**	**0.394**	**0.538**	**0.930**	**0.499**	**0.649**	**0.874**	**0.448**	**0.589**

Table 9. **Large-scale top-100 predictions** with the same settings in Table 1.

Different LLaMA Versions. In Table 7, we compare two truncated versions of LLaMA, namely 1.78B models of LLaMA 1 [112] and LLaMA 2 [113]. LLaMA 2 marginally outperforms LLaMA 1 trained on G3M, and has comparable results trained on G70M.

Ranking Predictions. Our one-shot sampling method selects the final top-k labels based on the probabilities of their initial tokens. Table 8 demonstrates the effectiveness of this approach compared to using full label probabilities. Further details on ranking strategies can be found in A.2.

Large-scale Prediction. We evaluate our method on large-scale prediction, i.e., top-100 predictions, with the same settings as in Table 1. Table 9 shows our method's consistent ability to predict diverse labels as the number of predictions increases, where R and F_1 are improved, and P is decreased. Besides, CLIP [93] has a similar trend, but its performance is much lower than ours. Further, with inflating its gallery from base to the extended one, CLIP has a performance drop across all datasets, also observed in [19].

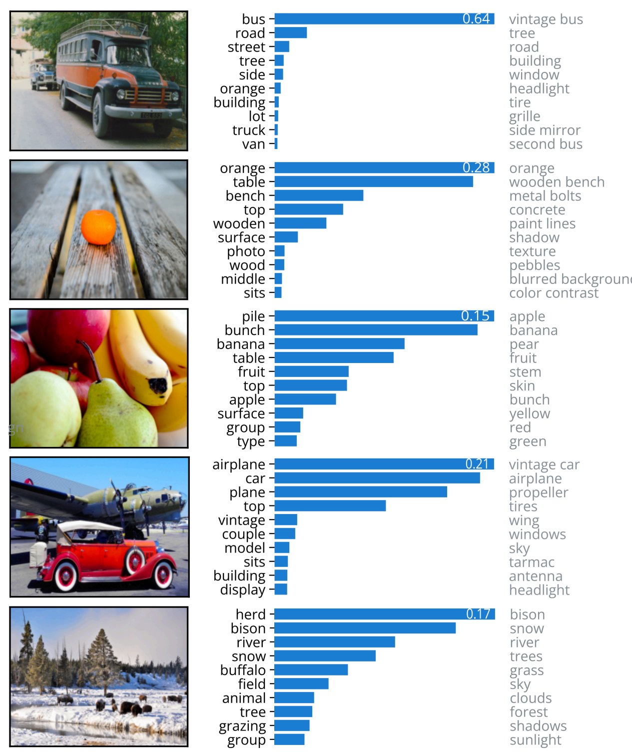


Figure 6. **Qualitative results** with top-10 predictions. The top bar is with the first prediction's probability. The right gray column displays GPT-4V Preview [86]'s predictions. For extensive results of 336 images, refer to Section A.8.

6. Conclusion

We have presented an auto-regressive framework for object recognition based on next token prediction, efficiently generating labels with one-shot sampling in parallel and intuitively depending only on the number of required labels.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: A Visual Language Model for Few-Shot Learning. In *NeurIPS*, 2022. 1, 2, 17

[2] Jacob Andreas and Dan Klein. Reasoning About Pragmatics With Neural Listeners and Speakers. In *EMNLP*, 2016. 2

[3] Anas Awadalla, Irena Gao, Josh Gardner, Jack Hessel, Yusuf Hanafy, Wanrong Zhu, Kalyani Marathe, Yonatan Bitton, Samir Gadre, Shiori Sagawa, et al. OpenFlamingo: An Open-Source Framework for Training Large Autoregressive Vision-Language Models. In *arXiv:2308.01390*, 2023. 1, 5, 6, 13, 16, 17

[4] Kobus Barnard and David Forsyth. Learning the Semantics of Words and Pictures. In *ICCV*, 2001. 18

[5] Kobus Barnard, Pinar Duygulu, David Forsyth, Nando De Freitas, David M Blei, and Michael I Jordan. Matching Words and Pictures. In *JMLR*, 2003. 18

[6] Abhijit Bendale and Terrance Boult. Towards Open World Recognition. In *CVPR*, 2015. 2

[7] Rodrigo Benenson and Vittorio Ferrari. From Colouring-in to Pointillism: Revisiting Semantic Segmentation Supervision. *arXiv:2210.14142*, 2022. 4

[8] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing With Python: Analyzing Text With the Natural Language Toolkit*. O'Reilly Media, Inc., 2009. 16

[9] David M Blei and Michael I Jordan. Modeling Annotated Data. In *ACM SIGIR*, 2003. 2

[10] Damian Borth, Rongrong Ji, Tao Chen, Thomas Breuel, and Shih-Fu Chang. Large-Scale Visual Sentiment Ontology and Detectors Using Adjective Noun Pairs. In *ACM MM*, 2013. 2

[11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language Models Are Few-Shot Learners. In *NeurIPS*, 2020. 1, 2, 3

[12] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12M: Pushing Web-Scale Image-Text Pre-training to Recognize Long-Tail Visual Concepts. In *CVPR*, 2021. 18

[13] Minmin Chen, Alice Zheng, and Kilian Weinberger. Fast Image Tagging. In *ICML*, 2013. 18

[14] Ting Chen, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. Pix2Seq: A Language Modeling Framework for Object Detection. In *ICLR*, 2022. 2

[15] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO Captions: Data Collection and Evaluation Server. *arXiv:1504.00325*, 2015. 4

[16] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90% ChatGPT Quality. *https://lmsys.org/blog/2023-03-30-vicuna*, 2023. 6, 16

[17] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling Instruction-Finetuned Language Models. *arXiv:2210.11416*, 2022. 6, 16

[18] Elijah Cole, Oisin Mac Aodha, Titouan Lorieul, Pietro Perona, Dan Morris, and Nebojsa Jojic. Multi-Label Learning From Single Positive Labels. In *CVPR*, 2021. 18

[19] Alessandro Conti, Enrico Fini, Massimiliano Mancini, Paolo Rota, Yiming Wang, and Elisa Ricci. Vocabulary-Free Image Classification. In *NeurIPS*, 2023. 1, 2, 5, 6, 8, 13, 16, 18

[20] Thomas Cover and Peter Hart. Nearest Neighbor Pattern Classification. *IEEE Trans. Inf. Theory*, 1967. 18

[21] Claudio Cusano, Gianluigi Ciocca, and Raimondo Schettini. Image Annotation Using SVM. In *SPIE*, 2003. 18

[22] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. InstructBLIP: Towards General-Purpose Vision-Language Models With Instruction Tuning. In *NeurIPS*, 2023. 2, 5, 6, 13, 16, 17

[23] Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. Generating Typed Dependency Parses From Phrase Structure Parses. In *LREC*, 2006. 2

[24] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum Likelihood From Incomplete Data via the EM Algorithm. *J. R. Stat. Soc. Ser. B (Methodol.)*, 1977. 18

[25] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009. 1, 3

[26] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NACCL-HLT*, 2019. 1

[27] Shizhe Diao, Wangchunshu Zhou, Xinsong Zhang, and Jiawei Wang. Write and Paint: Generative Vision-Language Models Are Unified Modal Learners. In *ICLR*, 2022. 3

[28] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*, 2021. 1, 2, 3, 4

[29] Yu Du, Fangyun Wei, Zihe Zhang, Miaojing Shi, Yue Gao, and Guoqi Li. Learning to Prompt for Open-Vocabulary Object Detection With Vision-Language Model. In *CVPR*, 2022. 2

[30] Thibaut Durand, Nazanin Mehrasa, and Greg Mori. Learning a Deep Convnet for Multi-Label Classification With Partial Labels. In *CVPR*, 2019. 18

[31] Pinar Duygulu, Kobus Barnard, Joao FG de Freitas, and David A Forsyth. Object Recognition as Machine Translation: Learning a Lexicon for a Fixed Image Vocabulary. In *ECCV*, 2002. 18

[32] Shao Lei Feng, Raghavan Manmatha, and Victor Lavrenko. Multiple Bernoulli Relevance Models for Image and Video Annotation. In *CVPR*, 2004. 18

[33] Rob Fergus, Yair Weiss, and Antonio Torralba. Semi-supervised Learning in Gigantic Image Collections. In *NeurIPS*, 2009. 18

[34] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. DeViSE: A Deep Visual-Semantic Embedding Model. In *NeurIPS*, 2013. 2

[35] Zihao Fu, Wai Lam, Anthony Man-Cho So, and Bei Shi. A Theoretical Analysis of the Repetition Problem in Text Generation. In *AAAI*, 2021. 2, 4

[36] Golnaz Ghiasi, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. Scaling Open-Vocabulary Image Segmentation With Image-Level Labels. In *ECCV*, 2022. 2

[37] Yunchao Gong, Yangqing Jia, Thomas Leung, Alexander Toshev, and Sergey Ioffe. Deep Convolutional Ranking for Multilabel Image Annotation. In *ICLR*, 2014. 18

[38] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-Vocabulary Object Detection via Vision and Language Knowledge Distillation. In *ICLR*, 2022. 2

[39] Matthieu Guillaumin, Thomas Mensink, Jakob Verbeek, and Cordelia Schmid. TagProp: Discriminative Metric Learning in Nearest Neighbor Models for Image Auto-Annotation. In *ICCV*, 2009. 18

[40] Tanmay Gupta, Arash Vahdat, Gal Chechik, Xiaodong Yang, Jan Kautz, and Derek Hoiem. Contrastive Learning for Weakly Supervised Phrase Grounding. In *ECCV*, 2020. 2

[41] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality Reduction by Learning an Invariant Mapping. In *CVPR*, 2006. 2

[42] Chi Han, Hengzhi Pei, Xinya Du, and Heng Ji. Zero-Shot Classification by Logical Reasoning on Natural Language Explanations. In *ACL*, 2023. 2

[43] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016. 1, 3

[44] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked Autoencoders Are Scalable Vision Learners. In *CVPR*, 2022. 6

[45] Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating Visual Explanations. In *ECCV*, 2016. 2

[46] Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. Grounding Visual Explanations. In *ECCV*, 2018. 2

[47] Tomer Hertz, Aharon Bar-Hillel, and Daphna Weinshall. Learning Distance Functions for Image Retrieval. In *CVPR*, 2004. 18

[48] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. In *Neural Computation*, 1997. 2

[49] Thomas Hofmann. Unsupervised Learning by Probabilistic Latent Semantic Analysis. *Machine Learning*, 2001. 2, 18

[50] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The Curious Case of Neural Text Degeneration. In *ICLR*, 2020. 2

[51] Xinyu Huang, Youcai Zhang, Jinyu Ma, Weiwei Tian, Rui Feng, Yuejie Zhang, Yaqian Li, Yandong Guo, and Lei Zhang. Tag2Text: Guiding Vision-Language Model via Image Tagging. In *ICLR*, 2024. 18

[52] Jiwoon Jeon, Victor Lavrenko, and Raghavan Manmatha. Automatic Image Annotation and Retrieval Using Cross-Media Relevance Models. In *ACM SIGIR*, 2003. 18

[53] Yangqing Jia, Mathieu Salzmann, and Trevor Darrell. Learning Cross-Modality Similarity for Multinomial Data. In *ICCV*, 2011. 18

[54] Thorsten Joachims. Optimizing Search Engines Using Clickthrough Data. In *ACM SIGKDD*, 2002. 18

[55] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. Densecap: Fully Convolutional Localization Networks for Dense Captioning. In *CVPR*, 2016. 2

[56] Andrej Karpathy and Li Fei-Fei. Deep Visual-Semantic Alignments for Generating Image Descriptions. In *CVPR*, 2015. 2

[57] Andrej Karpathy, Armand Joulin, and Li F Fei-Fei. Deep Fragment Embeddings for Bidirectional Image Sentence Mapping. In *NeurIPS*, 2014. 2

[58] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. CTRL: A Conditional Transformer Language Model for Controllable Generation. *arXiv:1909.05858*, 2019. 7

[59] Ryan Kiros, Ruslan Salakhutdinov, and Rich Zemel. Multimodal Neural Language Models. In *ICML*, 2014. 2

[60] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet Classification With Deep Convolutional Neural Networks. In *NeurIPS*, 2012. 1

[61] Weicheng Kuo, Yin Cui, Xiuye Gu, AJ Piergiovanni, and Anelia Angelova. F-VLM: Open-Vocabulary Object Detection Upon Frozen Vision and Language Models. In *ICLR*, 2023. 2

[62] Victor Lavrenko, Raghavan Manmatha, and Jiwoon Jeon. A Model for Learning the Semantics of Pictures. In *NeurIPS*, 2003. 18

[63] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep Learning. *Nature*, 2015. 18

[64] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. In *ICML*, 2022. 2, 4

[65] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping Language-Image Pre-training With Frozen Image Encoders and Large Language Models. In *ICML*, 2023. 2, 5, 6, 13, 16, 17

[66] Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In *ACL*, 2004. 5

[67] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014. 4

[68] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved Baselines with Visual Instruction Tuning. *arXiv:2310.03744*, 2023. 2, 5, 6, 13, 16, 17

[69] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual Instruction Tuning. In *NeurIPS*, 2023. 2, 5, 6, 13, 16, 17

[70] Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. Generating Wikipedia by Summarizing Long Sequences. In *ICLR*, 2018. 3

[71] Shilong Liu, Lei Zhang, Xiao Yang, Hang Su, and Jun Zhu. Query2Label: A Simple Transformer Way to Multi-Label Classification. *arXiv:2107.10834*, 2021. 18

[72] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A Convnet for the 2020s. In *CVPR*, 2022. 1

[73] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic Gradient Descent With Warm Restarts. In *ICLR*, 2017. 4

[74] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2019. 4

[75] Ziqiao Ma, Jiayi Pan, and Joyce Chai. World-To-Words: Grounded Open Vocabulary Acquisition Through Fast Mapping in Vision-Language Models. In *ACL*, 2023. 2

[76] Ameesh Makadia, Vladimir Pavlovic, and Sanjiv Kumar. A New Baseline for Image Annotation. In *ECCV*, 2008. 18

[77] Chengzhi Mao, Revant Teotia, Amrutha Sundar, Sachit Menon, Junfeng Yang, Xin Wang, and Carl Vondrick. Doubly Right Object Recognition: A Why Prompt for Visual Rationales. In *CVPR*, 2023. 2

[78] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, and Alan L Yuille. Explain Images With Multimodal Recurrent Neural Networks. *arXiv:1410.1090*, 2014. 2

[79] Marjo Markkula and Eero Sormunen. End-User Searching Challenges Indexing Practices in the Digital Newspaper Photo Archive. *Information Retrieval*, 2000. 18

[80] Sachit Menon and Carl Vondrick. Visual Classification via Description From Large Language Models. In *ICLR*, 2023. 2

[81] Jack Merullo, Louis Castricato, Carsten Eickhoff, and Ellie Pavlick. Linearly Mapping From Image to Text Space. In *ICLR*, 2023. 2

[82] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, et al. Simple Open-Vocabulary Object Detection. In *ECCV*, 2022. 2

[83] Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling Open-Vocabulary Object Detection. In *NeurIPS*, 2023. 2

[84] Florent Monay and Daniel Gatica-Perez. PLSA-Based Image Auto-Annotation: Constraining the Latent Space. In *ACM MM*, 2004. 18

[85] Yasuhide Mori, Hironobu Takahashi, and Ryuichi Oka. Image-To-Word Transformation Based on Dividing and Vector Quantizing Images With Words. In *First International Workshop on Multimedia Intelligent Storage and Retrieval Management*, 1999. 18

[86] OpenAI. GPT-4V(ision) System Card. *OpenAI Blog*, 2023. 5, 8, 13, 17, 18

[87] OpenAI. GPT-4 Technical Report. *arXiv:2303.08774*, 2023. 1, 2, 3

[88] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. Im2Text: Describing Images Using 1 Million Captioned Photographs. In *NeurIPS*, 2011. 4

[89] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *ACL*, 2002. 5

[90] Ofir Press and Lior Wolf. Using the Output Embedding to Improve Language Models. *EACL*, 2017. 3

[91] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving Language Understanding by Generative Pre-training. *OpenAI Blog*, 2018. 1, 2, 3

[92] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language Models Are Unsupervised Multitask Learners. *OpenAI Blog*, 2019. 1, 2, 3

[93] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*, 2021. 1, 2, 3, 4, 5, 6, 8, 13, 15, 16

[94] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the Limits of Transfer Learning With a Unified Text-To-Text Transformer. In *JMLR*, 2020. 3

[95] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-Shot Text-To-Image Generation. In *ICML*, 2021. 3

[96] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings Using Siamese Bert-Networks. In *EMNLP*, 2019. 5

[97] Paul Resnick and Hal R Varian. Recommender Systems. *ACM Communications*, 1997. 2

[98] Tal Ridnik, Gilad Sharir, Avi Ben-Cohen, Emanuel Ben-Baruch, and Asaf Noy. ML-Decoder: Scalable and Versatile Classification Head. In *WACV*, 2023. 18

[99] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. In *IJCV*, 2015. 15, 16

[100] Khalid Saifullah, Yuxin Wen, Jonas Geiping, Micah Goldblum, and Tom Goldstein. Seeing in Words: Learning to Classify Through Language Bottlenecks. In *ICLR Track on Tiny Papers*, 2023. 2

[101] Florian Schroff, Antonio Criminisi, and Andrew Zisserman. Harvesting Image Databases From the Web. *TPAMI*, 2010. 18

[102] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. LAION-400M: Open Dataset of Clip-Filtered 400 Million Image-Text Pairs. In *NeurIPS Workshop on Data-Centric AI*, 2021. 4

[103] Sarah Schwettmann, Neil Chowdhury, and Antonio Torralba. Multimodal Neurons in Pretrained Text-Only Transformers. In *ICCV Workshop on CLVL*, 2023. 2

[104] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual Captions: A Cleaned, Hypernymed, Image Alt-Text Dataset for Automatic Image Captioning. In *ACL*, 2018. 4, 5, 18

[105] Alex Sherstinsky. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. *Physica D - Nonlinear Phenomena*, 2020. 2

[106] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *ICLR*, 2015. 1

[107] Richard Socher and Li Fei-Fei. Connecting Modalities: Semi-supervised Segmentation and Annotation of Images Using Unaligned Text Corpora. In *CVPR*, 2010. 18

[108] Richard Socher, Andrej Karpathy, Quoc V Le, Christopher D Manning, and Andrew Y Ng. Grounded Compositional Semantics for Finding and Describing Images With Sentences. In *TACL*, 2014. 2

[109] Kihyuk Sohn. Improved Deep Metric Learning With Multi-Class N-Pair Loss Objective. In *NeurIPS*, 2016. 2

[110] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper With Convolutions. In *CVPR*, 2015. 1

[111] MosaicML NLP Team. Introducing MPT-7B: A New Standard for Open-Source, Commercially Usable LLMs. *https://www.mosaicml.com/blog/mpt-7b*, 2023. 6, 13, 16

[112] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. LLaMA: Open and Efficient Foundation Language Models. *arXiv:2302.13971*, 2023. 1, 2, 3, 6, 8, 16

[113] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. LLaMA 2: Open Foundation and Fine-Tuned Chat Models. *arXiv:2307.09288*, 2023. 2, 3, 6, 7, 8, 16

[114] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *NeurIPS*, 2017. 1, 2, 3

[115] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and Tell: A Neural Image Caption Generator. In *CVPR*, 2015. 2

[116] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. GIT: A Generative Image-To-Text Transformer for Vision and Language. In *TMLR*, 2022. 3

[117] Thomas Wang, Adam Roberts, Daniel Hesslow, Teven Le Scao, Hyung Won Chung, Iz Beltagy, Julien Launay, and Colin Raffel. What Language Model Architecture and Pretraining Objective Works Best for Zero-Shot Generalization? In *ICML*, 2022.

[118] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. SimVLM: Simple Visual Language Model Pretraining With Weak Supervision. In *ICLR*, 2022. 3

[119] Jason Weston, Samy Bengio, and Nicolas Usunier. Wsabie: Scaling up to Large Vocabulary Image Annotation. *Google*, 2011. 2

[120] Hu Xu, Saining Xie, Xiaoqing Ellen Tan, Po-Yao Huang, Russell Howes, Vasu Sharma, Shang-Wen Li, Gargi Ghosh, Luke Zettlemoyer, and Christoph Feichtenhofer. Demystifying CLIP Data. In *ICLR*, 2024. 1

[121] Jin Xu, Xiaojiang Liu, Jianhao Yan, Deng Cai, Huayang Li, and Jian Li. Learning to Break the Loop: Analyzing and Mitigating Repetitions for Neural Text Generation. In *NeurIPS*, 2022. 2, 4

[122] Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin, Chris Callison-Burch, and Mark Yatskar. Language in a Bottle: Language Model Guided Concept Bottlenecks for Interpretable Image Classification. In *CVPR*, 2023. 2

[123] Alireza Zareian, Kevin Dela Rosa, Derek Hao Hu, and Shih-Fu Chang. Open-Vocabulary Object Detection Using Captions. In *CVPR*, 2021. 2

[124] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. BERTScore: Evaluating Text Generation With Bert. In *ICLR*, 2020. 5, 16

[125] Youcai Zhang, Xinyu Huang, Jinyu Ma, Zhaoyang Li, Zhaochuan Luo, Yanchun Xie, Yuzhuo Qin, Tong Luo, Yaqian Li, Shilong Liu, et al. Recognize Anything: A Strong Image Tagging Model. *arXiv:2306.03514*, 2023. 18